 Annexe SQL 2025

Résumé complet avec exemples | Bac STI

 Types de données SQL

Types de données standards

```
INT
-- Entier (ex: 1, 42, 1000)
DECIMAL(10,2) -- Nombre décimal (ex: 123.45)
CHAR(10)
-- Chaîne de longueur fixe (ex: "AB" → "AB ")
VARCHAR(255)
-- Chaîne de longueur variable (ex: "Bonjour")
TEXT
-- Chaîne longue variable
DATE
-- Date (ex: '2025-06-06')
TIME
-- Heure (ex: '14:30:00')
DATETIME
-- Date et heure (ex: '2025-06-06 14:30:00')
```

 Opérateurs SQL

Opérateurs de comparaison

```
= -- Égal à
<>
-- Différent de
>
-- Supérieur à
```

```
<
-- Inférieur à
>=
-- Supérieur ou égal à
<=
-- Inférieur ou égal à
IN
-- Dans une liste de valeurs
BETWEEN
-- Entre deux valeurs
LIKE
-- Recherche de motif (% pour n'importe quelle chaîne, _ pour un
  caractère)
IS
-- Filtre les valeurs NULL
```

📌 **Exemples**

```
SELECT *
FROM produits
WHERE prix >
100; SELECT *
FROM clients
WHERE ville
IN ('Tunis',
'Ariana');
SELECT *
FROM produits
WHERE prix
BETWEEN 50
AND 100;
SELECT *
FROM produits
WHERE nom LIKE
'%phone%';
SELECT *
FROM produits
WHERE description
IS NULL;
```

Opérateurs logiques

```
AND
```

```
-- ET (les deux conditions doivent être vraies)
OR
-- OU (au moins une condition doit être vraie)
NOT
-- NON (inverse la condition)
```

📌 **Exemples**

```
SELECT *
FROM produits
WHERE prix >
50 AND stock
> 10;

SELECT *
FROM clients
WHERE ville =
'Tunis' OR ville
= 'Ariana';

SELECT *
FROM produits
WHERE NOT (prix
< 10);
```

 Fonctions sur les dates

Fonctions définies sur le type date

```
DATE()
-- Extrait la date d'une chaîne DATETIME
DAY()
-- Retourne le jour (1-31)
MONTH()
-- Retourne le mois (1-12)
YEAR()
-- Retourne l'année
NOW()
-- Retourne la date et l'heure courantes
ADDDATE()
-- Ajoute un intervalle de temps
```

```
DATEDIFF()
-- Différence entre deux dates en jours
```

📌 **Exemples**

```
SELECT
YEAR(date_commande)
FROM commandes;
SELECT MONTH(NOW()) AS mois_actuel;
SELECT
ADDDATE(NOW(),
7)
AS date_dans_7_jours;
SELECT
DATEDIFF(date_fin, date_debut)
AS duree
FROM projets;
```

📈 Fonctions d'agrégation

Fonctions d'agrégation

```
AVG()
-- Retourne la moyenne des valeurs
COUNT()
-- Retourne le nombre d'enregistrements
MAX()
-- Retourne la valeur maximale
MIN()
-- Retourne la valeur minimale
SUM()
-- Retourne la somme des valeurs
```

📌 **Exemples**

```
SELECT AVG(prix)
FROM produits;
```

```sql
SELECT COUNT(*)
FROM clients;
SELECT MAX(prix),
MIN(prix)
FROM produits;
SELECT categorie,
SUM(stock)
FROM produits
GROUP BY categorie;
```

 Manipulation des données (LMD)

SELECT - Interroger des données

```sql
SELECT [DISTINCT]
expression [, colonne, ... ou *] [[AS]
alias] FROM table1 [[AS] alias] [, table2, ...] [WHERE condition]
[GROUP BY critère] [HAVING
condition] [ORDER BY expression [ASC/DESC]]
```

📌 **Exemples**

```sql
-- Sélection simple
SELECT *
FROM produits;

-- Avec condition
SELECT nom, prix
FROM produits
WHERE prix >
50;

-- Avec alias
SELECT nom
AS produit, prix
AS montant
FROM produits;

-- Valeurs distinctes
SELECT
DISTINCT categorie
```

```sql
FROM produits;

-- Agrégation avec GROUP BY
SELECT categorie,
COUNT(*)
FROM produits
GROUP BY categorie;

-- Tri
SELECT *
FROM produits
ORDER BY prix
DESC;
```

INSERT INTO - Ajouter des données

```sql
INSERT INTO table [[liste colonnes]]
VALUES [liste valeurs]
```

📌 **Exemples**

```sql
-- Insertion avec toutes les colonnes
INSERT INTO produits
VALUES (1,
'Smartphone',
599.99, 10);

-- Insertion avec colonnes spécifiques
INSERT INTO produits (nom, prix, stock)
VALUES ('Tablette', 299.99, 5);

-- Insertion multiple
INSERT INTO produits (nom, prix)
VALUES ('Ordinateur', 899.99), ('Souris', 19.99);
```

UPDATE - Modifier des données

```
UPDATE table1
SET colonne = expression [WHERE
condition]
```

📌 **Exemples**

```
-- Mise à jour simple
UPDATE produits
SET prix =
649.99 WHERE id =
1;

-- Mise à jour avec condition
UPDATE produits
SET stock = stock -
1 WHERE id =
2 AND stock
> 0;

-- Mise à jour multiple colonnes
UPDATE produits
SET prix = prix *
1.1, stock = stock +
5 WHERE categorie
= 'électronique';
```

DELETE - Supprimer des données

```
DELETE FROM table1 [WHERE
condition]
```

📌 **Exemples**

```
-- Suppression avec condition
DELETE FROM produits
WHERE id = 5;
```

```
-- Suppression de tous les enregistrements
DELETE FROM produits;

-- Suppression avec sous-requête
DELETE FROM produits
WHERE prix
< (SELECT
AVG(prix)
FROM produits);
```

 Contraintes d'intégrité

Contraintes sur les colonnes

```
NOT NULL
-- Interdit une valeur nulle
DEFAULT
-- Attribue une valeur par défaut
CHECK
-- Spécifie une condition à respecter
PRIMARY KEY
-- Définit une clé primaire
UNIQUE
-- Spécifie l'unicité des valeurs
FOREIGN KEY
-- Définit une clé étrangère
REFERENCES
-- Fait référence à une clé primaire d'une autre table
ON UPDATE
-- Met à jour la clé étrangère
CASCADE
-- Supprime automatiquement les enregistrements liés
```

📌 **Exemple de création de table avec contraintes**

```
CREATE TABLE commandes ( id
INT
PRIMARY KEY
AUTO_INCREMENT, client_id
INT
NOT NULL, montant
```

```
DECIMAL(10,2)
DEFAULT
0, date_commande
DATE
DEFAULT
CURRENT_DATE, statut
VARCHAR(20)
CHECK (statut
IN ('en attente',
'expédiée',
'livrée')),
FOREIGN KEY (client_id)
REFERENCES clients(id)
ON UPDATE
CASCADE
ON DELETE
CASCADE
);
```

 Définition des données (LDD)

CREATE / DROP / ALTER

```
-- Bases de données
CREATE DATABASE nom_bd;
DROP DATABASE nom_bd;

-- Tables
CREATE TABLE table1 ( colonne1
type [contrainte], ... );
DROP TABLE table1;

-- Modification de table (ALTER)
ALTER TABLE table1
ADD colonne
type [contrainte];
ALTER TABLE table1
DROP colonne;
ALTER TABLE table1
MODIFY colonne
type;
-- Modifie le type
ALTER TABLE table1
CHANGE ancien_nom nouveau_nom
```

```
type;
-- Renomme une colonne

-- Contraintes
ALTER TABLE table1
ADD CONSTRAINT nom_contrainte
PRIMARY KEY (colonne);
ALTER TABLE table1
ADD CONSTRAINT nom_contrainte
FOREIGN KEY (colonne)
REFERENCES table2(colonne);
ALTER TABLE table1
DROP CONSTRAINT nom_contrainte;
ALTER TABLE table1
ENABLE/DISABLE CONSTRAINT
nom_contrainte;
```

📌 **Exemples complets**

```
-- Création d'une base de données
CREATE DATABASE boutique;

-- Création d'une table clients
CREATE TABLE clients ( id
INT
PRIMARY KEY
AUTO_INCREMENT, nom
VARCHAR(100)
NOT NULL, email
VARCHAR(255)
UNIQUE, date_inscription
DATE
DEFAULT
CURRENT_DATE
);

-- Ajout d'une colonne
ALTER TABLE clients
ADD telephone
VARCHAR(15);

-- Suppression d'une colonne
ALTER TABLE clients
DROP telephone;

-- Modification du type d'une colonne
ALTER TABLE clients
MODIFY nom
VARCHAR(150);

-- Ajout d'une contrainte
```

```sql
ALTER TABLE clients
ADD CONSTRAINT chk_email
CHECK (email
LIKE '%@%');

-- Suppression d'une table
DROP TABLE clients;
```

📝 Tableau récapitulatif des commandes SQL

Commande	Description	Exemple
SELECT	Interroger des données	SELECT * FROM produits;
INSERT INTO	Ajouter des données	INSERT INTO produits (nom, prix) VALUES ('PC', 999);
UPDATE	Modifier des données	UPDATE produits SET prix = 899 WHERE id = 1;
DELETE FROM	Supprimer des données	DELETE FROM produits WHERE id = 1;
CREATE TABLE	Créer une table	CREATE TABLE test (id INT, nom VARCHAR(50));
DROP TABLE	Supprimer une table	DROP TABLE test;
ALTER TABLE	Modifier une table	ALTER TABLE test ADD age INT;
CREATE DATABASE	Créer une base	CREATE DATABASE ma_base;
DROP DATABASE	Supprimer une base	DROP DATABASE ma_base;
GROUP BY	Grouper des résultats	GROUP BY categorie
ORDER BY	Trier des résultats	ORDER BY prix DESC
WHERE	Filtrer des résultats	WHERE prix > 100

🏆 Exemple complet - Gestion d'une boutique

```sql
-- 1. Création de la base de données
CREATE DATABASE boutique;
USE boutique;

-- 2. Création de la table clients
CREATE TABLE clients ( id
INT
```

```sql
PRIMARY KEY
AUTO_INCREMENT, nom
VARCHAR(100)
NOT NULL, email
VARCHAR(255)
UNIQUE, ville
VARCHAR(50) );

-- 3. Création de la table produits
CREATE TABLE produits ( id
INT
PRIMARY KEY
AUTO_INCREMENT, nom
VARCHAR(100)
NOT NULL, prix
DECIMAL(10,2), stock INT
DEFAULT 0
);

-- 4. Insertion de données
INSERT INTO clients (nom, email, ville)
VALUES ('Dupont Jean', 'jean@email.com',
'Tunis'), ('Ben Ali Sami', 'sami@email.com',
'Ariana');

INSERT INTO produits (nom, prix, stock)
VALUES ('Smartphone', 599.99, 10),
('Tablette',
299.99, 5);

-- 5. Requêtes d'interrogation
-- Liste des produits dont le prix est supérieur à 300€
SELECT *
FROM produits
WHERE prix >
300;

-- Statistiques par ville
SELECT ville,
COUNT(*)
AS nb_clients
FROM clients
GROUP BY ville;

-- Produits avec stock faible (moins de 10)
SELECT nom, stock
FROM produits
WHERE stock <
10 ORDER BY stock
ASC;

-- Mise à jour du stock après commande
UPDATE produits
SET stock = stock -
```

```sql
1 WHERE nom =
'Smartphone';

-- Suppression d'un client
DELETE FROM clients
WHERE id = 2;
```

 Annexe SQL 2025 - Lycée Cité Des Journalistes - Ariana
Conforme au programme officiel STI